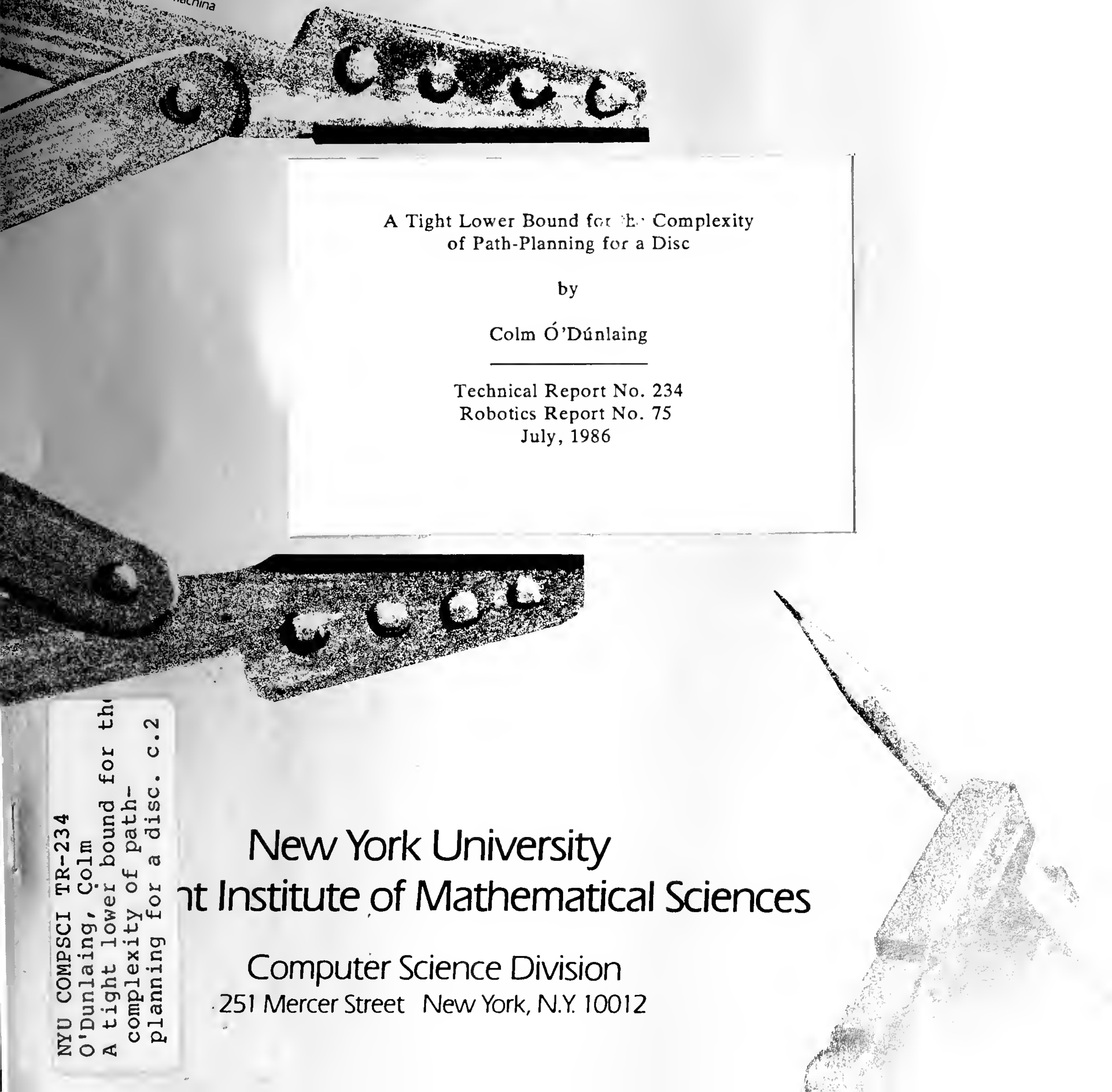
Robotics Research Technical Report



A Tight Lower Bound for the Complexity of Path-Planning for a Disc

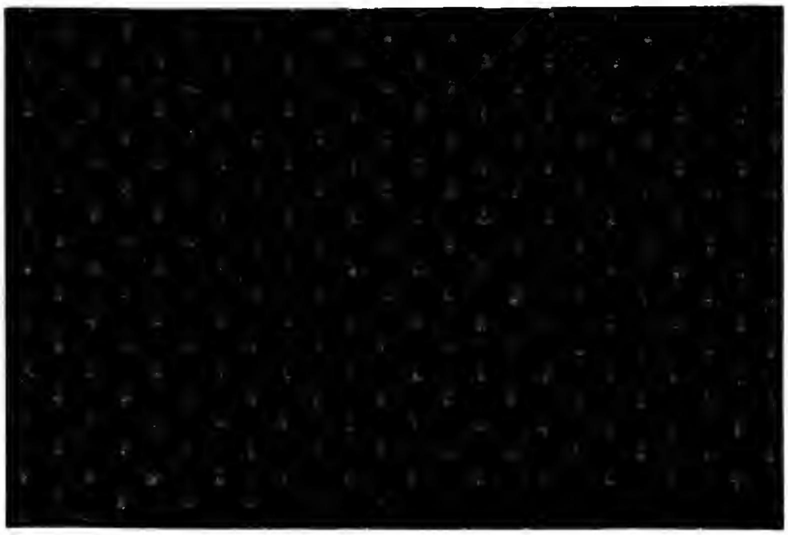
by

Colm Ó'Dúnlaing

Technical Report No. 234
Robotics Report No. 75
July, 1986

New York University
nt Institute of Mathematical Sciences

Computer Science Division
251 Mercer Street New York, N.Y. 10012



A Tight Lower Bound for the Complexity of Path-Planning for a Disc

by

Colm Ó'Dúnlaing

Technical Report No. 234
Robotics Report No. 75
July, 1986

New York University
Dept. of Computer Science
Courant Institute of Mathematical Sciences
251 Mercer Street
New York, New York 10012

This work was supported in part by the National Science Foundation under grant DCR 84-01898.

A tight lower bound for the complexity of path-planning for a disc.

Colm Ó'Dúnlaing

Courant Institute of Mathematical Sciences
251 Mercer Street
New York, New York 10012

ABSTRACT

Given two points in a planar room with polygonal boundary and obstacles, the problem of finding a *shortest* obstacle-avoiding path between them is known to require $\Omega(n\log(n))$ time. In this note it is shown that the problem of finding *any* obstacle-avoiding path for a disc in the room, or even deciding whether such a path exists, requires $\Omega(n\log(n))$ time. This bound is met by published algorithms.

A tight lower bound for the complexity of path-planning for a disc.

Colm Ó'Dúnlaing

Courant Institute of Mathematical Sciences
251 Mercer Street
New York, New York 10012

Keywords: computational geometry, motion-planning, complexity, algebraic decision tree.

1. Introduction.

Motion-planning problems are generally of the following flavor: Let Ω be a 'room,' i.e., a bounded open region in 2- or 3-dimensional Euclidean space, and suppose that B is a 'body' which can move within Ω subject to certain internal constraints on the geometry of B and the external constraint that B should not touch or penetrate the boundary of Ω; then given initial and final *placements* Z_0 and Z_1 of B within Ω, to determine whether it is possible to move B continuously from placement Z_0 to placement Z_1 subject to the given constraints, and, if so, to design one such feasible motion. The most general treatment of these problems as computational problems requires merely that the constraints on the motion be described by some piecewise algebraic relations, in which case, so long as the body B (which can incorporate several linkages and even disconnected parts) is specified in advance, one can solve the problem in time polynomial in the complexity of Ω. The paper by Schwartz and Sharir [7] demonstrates this by reducing the problem to Tarski geometry; typically the algorithms generated are impractical because of the size of the polynomial runtimes involved. On the other hand it is known that if the description of B is regarded as part of the input then the problem is NP-hard for moving several discs [8] or PSPACE-hard, for similar problems [2,3,6]. Generally the source of the complexity is the number of degrees of freedom which the body B possesses. In real motion-planning problems B might have six or seven degrees of freedom. O'Rourke [5] has shown that when the path-planning involves moving an oriented line-segment (a 'ladder') in a planar polygonal room of size n, the 'simplest' path can have a descriptive complexity of $\Omega(n^2)$.

A very simple instance of the motion-planning problem is when B is a disc and Ω is a planar room with polygonal boundary. (The room need not be simply connected: typically it has an enclosing wall and several fixed obstacles.) In this case it has been shown that $O(n\log(n))$ preprocessing time and $O(n)$ query time suffice (Ó'Dúnlaing and Yap [4]). In this note it will be shown that, under a reasonable formulation of what the output description of a planned path should be, the algorithm requires $\Omega(n\log(n))$ time in the worst case. This is achieved by reducing sorting to path-planning for a disc. Assuming an $\Omega(n\log(n))$ lower bound for the sorting problem, the result follows. It is to be emphasized that the $\Omega(n\log(n))$ bound, established in this note, is tight: in contrast, for instance, the shortest-path problem for a point in the plane is known to have complexity between $O(n\log(n))$ and $O(n^2)$ [9].

To assume the $\Omega(n\log(n))$ lower bound for sorting is not fully justifiable since it is not clear that the comparison model applies here. This criticism is addressed in the latter section of the paper, where a more rigorous analysis is made based on the algebraic computation tree model as described by Ben-Or [1]. There we shall see that the question of the existence of a path requires $\Omega(n\log(n))$ time in this model of computation, while still assuming that the input to the path-planning problem is well-formed.

2. Definitions and terms.

Throughout this note, Ω will be a bounded planar 'room' whose boundary is the union of finitely many finite bounded polygons; B will be a disc of radius r. The polygons are assumed to be simple and pairwise disjoint. (For computational purposes one can assume that a description of Ω consists of listing each polygonal component C of the boundary as a set of points (its corners) sorted cyclically, with the convention that the interior of Ω will always lie to the right of a flea traversing the polygon in the sorted direction.)

Conventionally, a *path* in the room Ω is a continuous mapping $\pi:[0,1]\to\Omega$, where $[0,1]$ is the closed unit interval. In a computational setting, we need a finite description of such a path, and characterize it as follows: the path is composed of a finite number of k *segments,* characterized by a subdivision of $[0,1]$ into k 'time intervals' $[t_i,t_{i+1}]$, $0=t_0<t_1< \cdots <t_k=1$, where

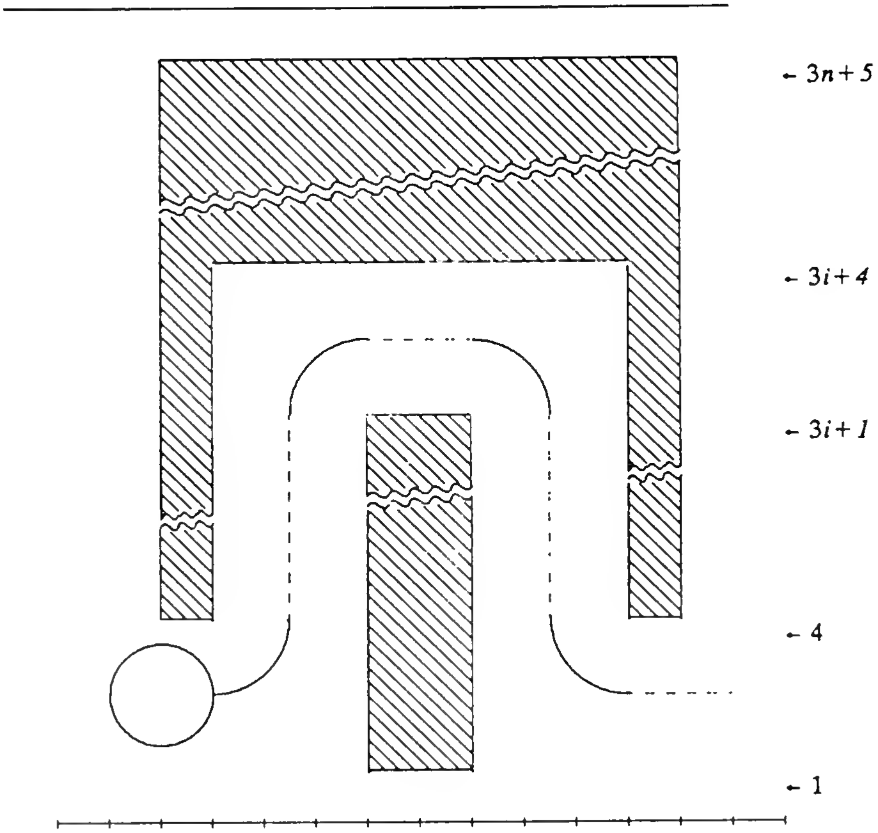
[1] π is continuously differentiable on each time-segment,

[2] for each interval, and for any direction $\mathbf{d}$ in the euclidean plane, the inner product $\mathbf{d}\cdot\dot{\pi}$ changes sign a bounded number of times, and the zeroes of this function (i.e., places where the velocity is normal to the direction $\mathbf{d}$) can be computed exactly in constant time for any interval. (Clearly the set of zeroes is a finite union of disjoint closed intervals.)

The endpoints t_j of the subintervals are called *nodes* of the path π. In the context of moving a disc B, we shall identify the path followed by B with the path followed by the center of the disc.

This characterization is wider than necessary: a more natural choice would be to require paths to be continuous and piecewise algebraic (with bounded degree on each piece), but conditions 1-3 above are strong enough to reduce the sorting problem to path-planning, as shown in the next section.

3. Reduction of sorting to motion-planning for a disc.



Illustrating how a disc can negotiate an (n,i)-barrier.

Suppose that we are given n real numbers

$$x_1, \cdots, x_n$$

which are not necessarily sorted, and suppose that we know that they are distinct and pairwise separated by a distance of at least 14 units. (We need to make this assumption since the path-planning problem assumes that the given obstacles do not overlap. A more rigorous presentation will be given in the next section.) We shall construct an instance of the path-planning problem such that a solution in the sense of the previous section can be transformed in linear time to a sorted listing of the given numbers. The reduction involves constructing what is essentially a corridor bounded by the $x-$axis and the line $y = 3n+6$. For each number x_i there is given a 'barrier' in this corridor, 10 units wide, centered about the vertical line $x = x_i$. The motion-planning problem is to move a disc of unit radius through all these barriers along the corridor. Every time the disc meets a barrier it must take a detour through a passageway sufficiently narrow to enable one to deduce the rank of x_i in sorted order.

Consider a fixed real number x; let i be an integer in the range $1 \cdots n$. Then an (n,i) *barrier* around x is constructed as illustrated. The barrier has two connected components. Its center component is a rectangle of width 2, centered on the vertical line through x, with its y-coordinates between 1 and $3i+1$. The other component has y-coordinates between 4 and $3n+5$, and is designed so that for the disc to pass the barrier it must negotiate a 'passage' around the center component, of width 3 units throughout, beginning with y-coordinate 4.

Lemma 1. Suppose that in traversing a path π the disc B of radius 1 crosses an (n,i) barrier around x_i. Then there is a point τ such that $\pi(\tau)$ has x-coordinate in the range (x_i-3,x_i+3) and y-coordinate in the range $(3i+2,3i+3)$, and either τ is a node of the path or $\mathbf{j}\cdot\dot{\pi}(\tau) = 0$, where $\mathbf{j}$ is the vector $(0,1)$.

Proof. In crossing the barrier it is necessary for the center of the disc B to cross the line $y=3i+2$ from below in the range (x_i-3,x_i-2) and from above in the range (x_i+2,x_i+3). Suppose that two such crossing points are at τ_1 and τ_2 respectively. If there is a node of π in the range (τ_1,τ_2) then there is nothing more to show. Otherwise consider the function $f(t) = \mathbf{j}\cdot\pi(t)$; by hypothesis it is continuous, and continuously differentiable in (τ_1,τ_2), and $f(\tau_1)=f(\tau_2)$. Therefore by Rolle's Theorem $\dot{f}$ has a zero in the given range. ●

Theorem 2. Suppose we are given a list $x_1 \cdots x_n$ of real numbers such that $|x_i-x_j|\geq 14$ for all pairs i and j of distinct subscripts. Then in $O(n)$ time we can construct a solvable $O(n)$ size problem for moving a disc, such that a solution path π with k nodes (in the sense of the previous section) can be used to sort the numbers x_i in $O(k)$ time. Consequently, in any computational model where the given sorting problem takes $\Omega(n\log(n))$ time, so does the motion-planning problem for a disc.

Proof. We assume that the numbers x_i are stored in an array so they are directly accessible through their subscripts. Let m and M be the minimum and maximum of the x_i respectively. A motion-planning problem is defined as follows. B is a disc of unit radius. Ω is a room contained in the rectangle $[m-9,M+9]\times[0,3n+6]$. For each i the room contains an (n,i)-barrier around x_i. Then the problem is to move the (center of the) disc B from $(m-7,2)$ to $(M+7,2)$.

Clearly this problem has size $O(n)$ and can be constructed in time $O(n)$. By the hypothesis that the numbers x_i are well separated it is clear that a solution path always exists.

Let π be a solution path with k nodes. Enumerate all points τ_j such that either τ_j is a node of the path or $\mathbf{J}\cdot\dot{\pi}(\tau_j) = 0$. By hypothesis there are $O(k)$ such points and they can be listed, in their order of occurrence, in time $O(k)$. Process the points τ_j in left-to-right order.

Since the path π is not necessarily monotonic in the x-direction, we need to use a pushdown stack to deduce the permutation σ required to sort the set of points x_i. The stack will hold subscripts in the range 1..n and is intially empty. For each point τ_j, let (x,y) be the coordinates of $\pi(\tau_j)$. If the floor of y is not of the form $3i+2$ for some $i \geq 1$, ignore the point τ_j. If it is, evaluate $|x-x_i|$; if this distance is greater than 3, ignore the point τ_j. Otherwise, τ_j is a point where the disc is negotiating the barrier around x_i.

If the stack is empty, or if $x_r < x_i$ where r is the subscript on top of the stack, then push i onto the stack. If $r=i$, do nothing, and if $x_r > x_i$ (so the disc is travelling backwards), discard r from the top of the stack. It is not difficult to see that this process maintains the invariant condition that the subscripts on the stack define an initial segment of the sorted sequence of points x_i, namely, those points to the left of the point being currently processed (each such point x_i must contribute at least one point τ_j by Lemma 1). Thus ultimately, when the disc reaches its destination, the stack contents define the permutation σ which sorts the numbers. This processing clearly takes $O(k)$ time.

Suppose that motion-planning for a problem of size $O(n)$ did not require $\Omega(n\log(n))$ time, so there was an algorithm generating solution paths for the problem instance described here in time $o(n\log(n))$. In particular the size k of the output description would also be $o(n\log(n))$, so we could implement the above scheme to sort the numbers x_i in time $O(n)+o(n\log(n)) = o(n\log(n))$, a contradiction. **Q.E.D.**

Remark. In the paper of Ó'Dúnlaing and Yap [4] it was pointed out that the motion-planning problem could be solved by constructing the '1-fringe' of the set of obstacles, namely, the set of points in the room whose distance from the closest wall or corner was exactly 1, but it was remarked that no $o(n\log(n))$ method of computing the fringe was known. The results of this paper explain this: indeed, it is clear that in the construction given above a description of the r-fringe could be used to sort the points x_i.

4. Lower bound in the algebraic computational model.

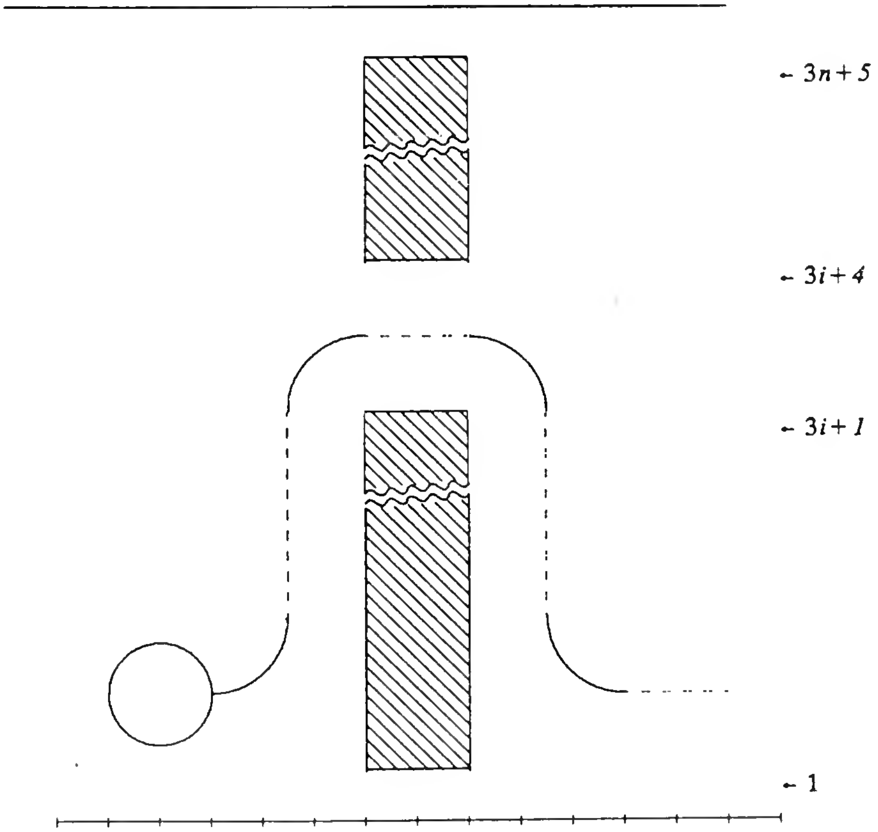
In this section we shall see that the motion-planning problem for a disc has complexity $\Omega(n\log(n))$ in the algebraic computation-tree model described by Ben-Or [1]. For a fixed positive integer n and nonegative real number r let

$$W_r = \{(x_1, \cdots, x_n) \in \mathbf{R}^n : \forall\, i \neq j\ (|x_i - x_j| > r)\}.$$

Thus W_0 represents the set of tuples of pairwise distinct real numbers. The following lemma is a straightforward extension of the argument given by Ben-Or for the set W_0.

Lemma 3. For any $r<s$ and fixed n, the set $W_r - W_s$ has exactly $n!$ (path) components, and every component is contained in a unique component of W_0. If p and q are points in different components of W_s then any path connecting them meets $W_r - W_\sigma$ somewhere, for every σ in $(r,s]$. ●

Given a list $(x_1, \cdots, x_n)$ as before, we construct another (and simpler) instance of the path-planning problem, such that there exists a solution if and only if the given tuple is in W_4. This time an (n,i) barrier is redefined simply as two rectangles, of width 2, symmetrically centered around the vertical line through x_i. The lower rectangle is bounded by the lines $y=1$ and $y=3i+1$, and the upper rectangle is bounded by the lines $y=3i+4$ and $y=3n+5$.



Illustrating an (n,i) barrier for the decision problem.

As in Theorem 2, we construct an instance of the motion-planning problem for a unit radius disc B; everything is as before except for the redefinition of the 'barriers.'

Theorem 4. Suppose that

$$(x_1, \cdots, x_n)$$

is a tuple of points in W_3, and the motion-planning problem is constructed as above. Then any algebraic computation tree T which solves the given instances of the motion-planning problem has depth $\Omega(n\log(n))$.

Sketch of proof. Notice that the assumption about the tuple being in W_3 ensures that the instance of the motion-planning problem is well-formed (i.e., the obstacles do not overlap). Clearly, a solution path π exists if and only if the minimal separation of adjacent barriers is greater than 4 units (otherwise the disc is unable to squeeze out of between some two adjacent barriers): i.e., existence of a solution path is equivalent to the given tuple being in W_4.

We continue with the notation used by Ben-Or [1]. Consider any 'YES'-leaf ℓ of the putative computation tree T. Let V be the set of tuples in $\mathbf{R}^n$ which cause a decision path leading to the leaf ℓ. Claim that every (path-) component of V is completely contained in a component of W_4: otherwise, by the above lemma, there exists a point p in $V \cap W_3 - W_4$, and that point defines a legitimate instance of the motion-planning problem with a negative answer, so T is incorrect. The rest of the argument follows from Lemma 3 above and Theorem 5 of [1]. •

Remark. The constructions in Theorems 2 and 4 are subject to the criticism that the obstacles constructed are not in general position. However, in Theorem 4 the point p discussed could be chosen to be outside $W_{3.5}$, say, and this would correspond to a set of obstacles which could be perturbed slightly if necessary to remove collinearities and cocircularities, while still fooling the algorithm. The construction in Theorem 2 depends on Lemma 1 for its succcss, but it is not difficult to show that Lemma 1 is still true when the 'barriers' are slightly perturbed, so Theorem 2 also does not depend on the existence of degeneracies.

5. References.

[1] Ben-Or, M (1983). Lower bounds for algebraic computation trees. *Proceedings of the 15th ACM STOC Symposium, 80-86.*

[2] Hopcroft, J, Joseph, D, and Whitesides, S (1982). On the movement of robot arms in 2-dimensional bounded regions. *Proceedings of the 23rd IEEE FOCS Symp., 280-289.*

[3] Hopcroft, J, Schwartz, J, and Sharir, M (1984). On the complexity of motion-planning for multiple independent objects: PSPACE-hardness of the warehouseman's problem. Technical report 103, Computer Science Dept, New York University.

[4] Ó'Dúnlaing, C, and Yap, C (1985). A 'retraction' method for planning the motion of a disc. *J. Algorithms* 6 *104-111.*

[5] O'Rourke, Joseph (1985). A lower bound on moving a ladder. Technical report, EECS Department, Johns Hopkins University, Baltimore, MD 21218.

[6] Reif, J (1979). Complexity of the movers' problem and generalizations. *Proc. 20th IEEE FOCS Symp., 421-427.*

[7] Schwartz, J, and Sharir, M (1983). On the piano-movers' problem: II. General techniques for computing topological properties of real algebraic manifolds. *Advances in Applied Mathematics* 4, *298-351.*

[8] Spirakis, P, and Yap, C (1984). Strong NP-hardness of moving many discs. *Information Processing Letters* 19 *55-59.*

[9] Welzl, E (1985). Constructing the visibility graph for n line-segments in $O(n^2)$ time. *Information Processing Letters* 20 *167-171.*